ZJU Summer 2024 Contest 9

Contest Analysis

Group C

07.16.2024

A. Back and Forth

Description

给一个序列 $a_1, a_2, \ldots, a_n$，依次把这些数加入一个双端队列，每次可以加到开头或结尾

最小化最终双端队列中相邻数的差的最大值，输出方案

A. Back and Forth

Solution

本来是单 $\log$ ，但是因为这场少一个稍微简单点的题所以弱化到了双 $\log$ 。
有各种双 $\log$ 做法，比较好的一种是二分答案后用 set 维护加入第 i 个数后，第 i 个数的另一侧的可能数的集合
然后加入 a_{i+1} ，就考虑这个数能放在哪边即可。如果两边都能放，就把 a_i 也加入到集合。

然后考虑优化这个做法，注意到另一侧能加入的数总是一个区间，可以不用 set 维护。
时间复杂度 $O(n \log a)$

B. Jump

Description

Alice 要从 0 位置到 D 位置，跳 n 步，第 i 步，设当前位置 x ，则会选择 $x - a_i, x, a + a_i$ 中离 D 更近的随机一个走过去。
对于每个 i ，求如果能任意修改 a_i ，是否存在一种让 Alice 到不了 D 位置的方案。

之后的讨论用与 D 的距离作为位置。
考虑维护 S_i 表示走第 i 步到第 n 步，最终能够到达 D 的位置集合。
记 T_i 表示走前 i 步后到达的位置，如果 $mex(S_{i+1}) <= T_{i-1}$ ，那么就可以通过修改 a_i 来让 Alice 跳到 $mex(S_{i+1})$ 的位置，从而最终到不了 D。

B. Jump
Solution

观察到 S_i 中大于 $mex(S_i)$ 的位置都不再会对后续 mex 产生影响，所以：
若 $|mex(S_{i+1}) - 1 - a_i| <= mex(S_{i+1})$ ，则 $mex(S_i) = mex(S_{i+1}) + a_i$
否则，$mex(S_i) = mex(S_{i+1})$
时间复杂度 $O(n)$

有长度为 n 的序列 a_i ，可以在这个序列上移动，从 i 移动到 j 会减少 $k \times |i - j|$ 的得分。此外移动到某个位置 x 会使得得分增加 a_x 。起点任意，不能重复移动到同一个点。

问到达至多 m 个点能得到的最大得分。

转化题意：
选择一段区间 $[l, r]$, 强制选择两个端点上的魔法石，再从区间内选择若干值（包括端点总共不超过 m 个），最终贡献为 $Sum - (r - l) \times k$。然后我们要让这个贡献最大化。
可以发现 l, r 端点强制选择对我们限制很大，考虑将这个求解方式转换一下。
选择一段区间 $[l, r]$，直接从区间内选择最多 m 个数，最终贡献为 $Sum - (r - l) \times k$。可以看出我们第一种求解形式的贡献肯定是被第二种求解形式包含，而且由于 $k > 0$ 那么第二种求解形式选择的每段区间 $[l, r]$ 都能更优地缩成第一种求解形式。所以两种求解方式是等价的。

考虑第二种求解方式。如果我们已经有一段区间 $[l, r]$，显然我们会选取其中最大的 m 个数，可以使用主席树单 $\log$ 进行求解，可以证明决策单调性，对于每个 r，$best_l(r) <= best_l(r+1)$ 。这样我们可以通过分治求解每个 r 的决策点，然后继续递归下去即可。总共时间复杂度 $O(n \log^2 n)$ 。

D. Neatness of String

Description

定义一个字符串的 neatness 为：任意排序它的所有前缀后，相邻前缀的公共 border 长度之和的最大值。
给出字符串 S 和 m 个修改，每个修改为在最后添加某个字符或者删除末尾字符，每次操作后求 S 的 neatness 。

将每个前缀跟它的 border 连边，形成一棵树。这个部分可以通过计算 $t_{i,j}$ 表示在前缀 i 后面添加字符 j 后得到的 border 来维护。
两个前缀的公共 border 即为二者的 LCA 。使得这个东西长度和最大的排列方式为按 dfs 序排列。
离线 dfs 求出 dfs 序即可。但是看起来考场上的各位有更简便的在线维护方法。
时间复杂度 $O((n+m)|\Sigma| + (n+m)\log)$

$a_1 = 1, a_2 = 2, a_n = a_{n-1} + a_{n-2}$ ，称一个项 a_i 为 Single ，当且仅当 $i > 1$ 且不存在 $1 < j < i$ 满足 $a_j | a_i$ 。
多组询问，问 a_l 到 a_r 中有多少个 Single 。

给这个序列加几项，变成 $b_0 = 0, b_1 = 1, b_n = b_{n-1} + b_{n-2}$ 。
发现 b_i 为 Single 当且仅当 i 为奇素数或 4 。
证明：b_i 的倍数出现在 b_{ki} 的位置（可以通过 b_i 和 b_{i+1} 互素来证明），然后模拟一下埃氏筛
设下标值域 $[1, V]$ ，时间复杂度 $O(V + Q)$

F. Tree Between Trees

Description

给出两棵点数分别为 n, m 的树 T_1, T_2 ，在每一对属于不同树的点之间连边，求生成树个数。

设先完成了两棵树内的连边，T_1 连成 n' 个连通块，T_2 连成 m' 个连通块，那么这种情况下的方案树为 $\prod(\text{连通块大小}) \times n^{m'-1}m^{n'-1}$ 。证明如下：

首先考虑一个完全二分图 $K_{n,m}$ 的生成树问题，考虑其 prufer 序列的构造过程，由于最后留下的边一定连接两侧的点，所以共有 $n-1$ 个左侧点和 $m-1$ 个右侧点被删除。

于是 prufer 序列中，左侧点的子序列和右侧点的子序列的方案数分别为 n^{m-1} 和 m^{n-1} 。又因为在确定 prufer 序列的前 $i-1$ 项和每个点的度数以后，prufer 序列的第 i 项来自哪一边其实是确定的，所以方案数就是 $n^{m-1}m^{n-1}$。

然后回到这个问题，考虑连通块构成的完全二分图，在它的 prufer 序列的构成中，会删去 $n'-1$ 个左侧连通块和 $m'-1$ 个右侧连通块。删去某一侧连通块时，加入 prufer 序列的可以时任意一个点，所以总方案数为 $n^{m'-1}m^{n'-1}$ 。
然后考虑每个连通块被删除时，连出去的边的在连通块内的端点也是可以任选的，加上最后留下的一条边的方案数，一共是 $\prod(\text{连通块个数})$
所以总方案数为 $\prod(\text{连通块大小}) \times n^{m'-1}m^{n'-1}$ 。

所以总方案数为 $\prod(\text{连通块大小}) \times n^{m'-1} m^{n'-1}$ 。
观察这个式子，对于两棵树其实是独立的。$\prod(\text{连通块大小})$ 是经典技巧，转化为每个连通块内选一个关键点的方案数。
然后对两棵树分别 dp 即可，总时间复杂度 $O(n+m)$

Thanks!